

[illegible]

(1) 77
(1) 111
(1) 261
(1) 286

DECLARATIONS
MP\$QAST - ENQUEUE AST CONTROL BLOCK FOR PROCESS
MP\$ASTSCHEDCHK - CHECK FOR RESCHEDULE AT AST DELIVERY
MP\$ASTNEWLVL - CHECK FOR SETTING NEW ASTLVL


```
0000 1 :  
0000 2 : Version: 'V04-000'  
0000 3 :  
0000 4 :  
0000 5 : .MCALL MFPR  
0000 6 : .TITLE MPAST - MULTIPROCESSOR AST ROUTINES  
0000 7 : .IDENT 'V04-000'  
0000 8 :  
0000 9 : *****  
0000 10 : *  
0000 11 : * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY *  
0000 12 : * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS. *  
0000 13 : * ALL RIGHTS RESERVED. *  
0000 14 : *  
0000 15 : * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED *  
0000 16 : * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE *  
0000 17 : * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER *  
0000 18 : * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY *  
0000 19 : * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY *  
0000 20 : * TRANSFERRED. *  
0000 21 : *  
0000 22 : * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE *  
0000 23 : * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT *  
0000 24 : * CORPORATION. *  
0000 25 : *  
0000 26 : * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS *  
0000 27 : * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL. *  
0000 28 : *  
0000 29 : *****  
0000 30 :  
0000 31 : ++  
0000 32 :  
0000 33 : Facility: Executive, Scheduler  
0000 34 :  
0000 35 : Abstract: Primitives for AST queueing and delivery.  
0000 36 :  
0000 37 : Environment: MODE=Kernel  
0000 38 :  
0000 39 : Author: RICHARD I. HUSTVEDT, Creation date: 15-MAY-1979  
0000 40 :  
0000 41 : Modified by:  
0000 42 :  
0000 43 : V02-009 KDM0032 Kathleen D. Morse 27-Aug-1981  
0000 44 : Don't do rescheduling AST if process can not run on  
0000 45 : secondary due to non-zero PHDSL_MPINHIBIT.  
0000 46 :  
0000 47 : V02-008 KDM0015 Kathleen D. Morse 12-Jun-1981  
0000 48 : Add use of hook MPH$EMPTYCONT.  
0000 49 :  
0000 50 : V02-007 KDM0011 Kathleen D. Morse 29-Apr-1981  
0000 51 : Integrate a new performance-improved AST delivery routine.  
0000 52 : Use new exec labels for hooks: MPH$<name>.  
0000 53 :  
0000 54 : V02-006 KDM0010 Kathleen D. Morse 06-Apr-1981
```

```

0000 53 : Only cause rescheduling AST if there is some other
0000 54 : process to run (not null) and there are no AST's
0000 55 : queued to this process, in addition to other checks.
0000 56 :
0000 57 : V02-005 KDM0009 Kathleen D. Morse 06-Apr-1981
0000 58 : Add new routines, MP$ASTNEWLVL and MP$ASTSCHEDCHK,
0000 59 : that used to be in module, MPSCHED. Rename module
0000 60 : to be MPAST.
0000 61 :
0000 62 : V02-004 KDM0008 Kathleen D. Morse 01-Apr-1981
0000 63 : Set ASTLVL processor register to executive mode if
0000 64 : the secondary is idle and the PHD ASTLVL > PR ASTLVL.
0000 65 :
0000 66 : V02-003 KDM0003 Kathleen D. Morse 15-Sep-1980
0000 67 : Make changes to run on VMS V2.0.
0000 68 :
0000 69 : V02-002 KDM0001 Kathleen D. Morse 04-Sep-1980
0000 70 : Replace copyright.
0000 71 :
0000 72 : 01 -
0000 73 : --
0000 74 :
0000 75 :

```



```
0000 77      .SBTTL  DECLARATIONS
0000 78
0000 79  :
0000 80  : INCLUDE FILES:
0000 81  :
0000 82  :
0000 83  :
0000 84  : MACROS:
0000 85  :
0000 86  :
0000 87  :
0000 88  : EQUATED SYMBOLS:
0000 89  :
0000 90
0000 91      $ACBDEF      ; AST control block definitions
0000 92      $IPLDEF      ; IPL definitions
0000 93      $LCKDEF      ; Interlock bit definitions
0000 94      $MPSDEF      ; Secondary processor states
0000 95      $PCBDEF      ; PCB definitions
0000 96      $PHDDEF      ; PHD definitions
0000 97      $PRDEF       ; Processor register definitions
0000 98      $PRIDEF      ; Priority increment class defs
0000 99      $PSLDEF      ; PSL field definitions
0000 100     $SSDEF       ; Status code definitions
0000 101     $STATEDEF    ; Define state values
0000 102
00000000 0000 103  ASTEXIT = 0      ; AST exit change mode code
0000 104
0000 105  :
0000 106  : OWN STORAGE:
0000 107  :
00000000 108      .PSECT  A$EXENONPAGED, LONG
0000 109
```

```
0000 111 .SBTTL MPSSQAST - ENQUEUE AST CONTROL BLOCK FOR PROCESS
0000 112
0000 113 :++
0000 114 :
0000 115 : FUNCTIONAL DESCRIPTION:
0000 116 :
0000 117 : MPSSQAST inserts the AST control block supplied in the proper
0000 118 : position by access mode in the AST queue of the process specified
0000 119 : by the PID field of the AST control block. An AST arrival event
0000 120 : is then reported for the process to reactivate from a wait state
0000 121 : if appropriate. The AST control block will be released immediately
0000 122 : if the PID specifies a non-existent process.
0000 123 :
0000 124 : ENVIRONMENT:
0000 125 :
0000 126 : Executed by the primary processor.
0000 127 :
0000 128 : CALLING SEQUENCE:
0000 129 :
0000 130 : BSB/JSB MPSSQAST
0000 131 :
0000 132 : INPUT PARAMETERS:
0000 133 :
0000 134 : R2 - Priority increment class
0000 135 : R5 - Pointer to AST control block
0000 136 :
0000 137 : IMPLICIT INPUTS:
0000 138 :
0000 139 : PCB of process identified by PID field
0000 140 :
0000 141 : OUTPUT PARAMETERS:
0000 142 :
0000 143 : R0 - Completion status code
0000 144 : R4 - PCB address of process for which AST was queued
0000 145 :
0000 146 : SIDE EFFECTS:
0000 147 :
0000 148 : The process identified by the PID in the AST control block
0000 149 : will be made executable if not suspended.
0000 150 :
0000 151 : COMPLETION CODES:
0000 152 :
0000 153 : SS$_NORMAL - Normal successful completion status
0000 154 : SS$_NONEXPR - Non-existent process
0000 155 :
0000 156 : --
0000 157 :
0000 158 : .ENABL LSB
0000 159 QNONEXPR:
0000 160 MOVL R5,R0 : RELEASE AST CONTROL BLOCK
0003 161 JSB G^EXE$DEANONPAGED : IF NO SUCH PROCESS
0009 162 MOVZWL #SS$_NONEXPR,R0 : SET ERROR STATUS CODE
000E 163 BRB QEXIT : AND EXIT
0010 164
0010 165 MPSSQAST::
0010 166 MOVZWL ACB$_PID(R5),R0 : ENQUEUE AST FOR PROCESS
0014 167 DSBINT #IPL$_SYNCH : GET PROCESS INDEX FOR AST TARGET
: DISABLE SYSTEM EVENTS
```



```
54 00000000'GF D0 001A 168 MOVL G^SCH$GL_PCBVEC,R4 ; GET ADDRESS OF VECTOR
      54 6440 D0 0021 169 MOVL (R4)[R0],R4 ; LOOK UP PCB ADDRESS
60 A4 0C A5 D1 0025 170 CMPL ACB$B_PID(R5),PCB$B_PID(R4) ; CHECK FOR MATCH IN PID
      D4 12 002A 171 BNEQ QNONE$PR ; PID MISMATCHES
      50 D4 002C 172 CLRL R0 ; ASSUME KERNEL MODE AND CLEAR HIGH BITS
      10 A4 65 0E 002E 173 INSQUE (R5),PCB$B_ASTQFL(R4) ; ASSUME QUEUE IS EMPTY AND ATTEMPT INSERT
      3B 12 0032 174 BNEQ 50$ ; BR IF IT WAS NOT EMPTY
      0B A5 95 0034 175 TSTB ACB$B_RMOD(R5) ; CHECK FOR SPECIAL KERNEL AST
      06 19 0037 176 BLSS 10$ ; BR IF YES
50 0B A5 FC 8F 8B 0039 177 BICB3 #^C<3>,ACB$B_RMOD(R5),R0 ; GET AST MODE
      003F 178 ;
      003F 179 ; THE PROCESS HEADER ADDRESS IS ALWAYS A SYSTEM SPACE ADDRESS (NEGATIVE NUMBER)
      003F 180 ; WHILE THE PROCESS HEADER IS RESIDENT. DURING THE OUTSWAP TRANSITION IT IS
      003F 181 ; THE BALANCE SLOT INDEX, A SMALL POSITIVE NUMBER. FINALLY, AFTER OUTSWAP IT
      003F 182 ; IS SET TO ZERO. HENCE, THE FOLLOWING TEST COMBINES THE FETCH OF THE PHD
      003F 183 ; ADDRESS WITH THE TEST FOR PROCESS RESIDENCE.
      003F 184 ;
      51 6C A4 D0 003F 185 10$: MOVL PCB$B_PHD(R4),R1 ; POINT TO PROCESS HEADER
      17 18 0043 186 BGEQ 25$ ; DON'T SET ASTLVL IF NOT RESIDENT
      00CF C1 50 90 0045 187 MOV B R0,PHD$B_ASTLVL(R1) ; SET ASTLVL IN PROCESS HEADER
      2C A4 0E B1 004A 188 20$: CMPW #SCH$C_CUR,PCB$B_STATE(R4) ; IS PROCESS A CURRENT PROCESS?
      0C 12 004E 189 BNEQ 25$ ; BR ON NO, NOT A CURRENT PROCESS
54 00000000'GF D1 0050 190 CMPL G^SCH$GL_CURPCB,R4 ; IS PROCESS ON PRIMARY?
      11 13 0057 191 BEQL 40$ ; YES
      FFA4' 30 0059 192 BSBW MPSS$INTSCND ; INTERRUPT SECONDARY PROCESSOR
      005C 193 .LIST MEB
      00000000'GF 16 005C 194 25$: RPT ETVT AST ; REPORT AST ARRIVAL
      00 0062 195 25$: JSB G^SCH$RSE ; REPORT AST ARRIVAL
      50 01 3C 0063 196 .BYTE EVT$AST ; EVENT CODE
      12 8E DA 0066 197 30$: MOVZWL #SS$NORMAL,R0 ; SET SUCCESS STATUS CODE
      05 0069 198 QEXIT: ENBINT ; ENABLE INTERRUPTS
      006A 199 RSB MTPR (SP)+,S^#PR$_IPL ; AND RETURN
      006A 200 ;
      006A 201 ; IF THE AST IS BEING ENQUEUED FOR THE CURRENT PROCESS, THEN THE REPORTING
      006A 202 ; OF THE AST EVENT CAN BE BYPASSED AND THE ASTLVL PROCESSOR REGISTER MUST BE
      006A 203 ; SET INSTEAD.
      006A 204 ;
      13 50 DA 006A 205 40$: MTPR R0,#PR$ASTLVL ; ALSO SET ASTLVL REGISTER
      F4 11 006D 206 BRB 30$ ;
      006F 207 ;
      006F 208 ; THE AST QUEUE WAS NOT EMPTY (ITS USUAL CONDITION) AND THE PROPER
      006F 209 ; POSITION FOR THE NEW AST MUST BE LOCATED. SINCE THE AST CONTROL
      006F 210 ; BLOCK HAS BEEN ERRONEOUSLY INSERTED ON THE QUEUE, IT MUST BE REMOVED
      006F 211 ; FIRST.
      006F 212 ;
      51 55 65 OF 006F 213 50$: REMQUE (R5),R5 ; ELSE CORRECT MISTAKE
      10 A4 DE 0072 214 MOVAL PCB$B_ASTQFL(R4),R1 ; POINT TO QUEUE HEADER
      53 61 D0 0076 215 MOVL (R1),R3 ; AND GET FIRST ENTRY
      0B A5 95 0079 216 TSTB ACB$B_RMOD(R5) ; CHECK FOR SPECIAL KERNEL AST
      0F 18 007C 217 BGEQ 70$ ; BR IF NOT
      007E 218 ;
      007E 219 ; THE NEW AST IS A SPECIAL KERNEL AST. IT WILL GO AFTER ALL OTHER SPECIAL
      007E 220 ; KERNEL ASTS OR AT THE HEAD OF THE QUEUE IF THERE ARE NONE.
      007E 221 ;
      53 51 D1 007E 222 60$: CMPL R1,R3 ; CHECK FOR END OF QUEUE
      27 13 0081 223 BEQL 110$ ; BR IF NOT
```



```

      OB A3 95 0083 224      TSTB ACB$B_RMOD(R3)      : CHECK FOR SPECIAL KERNEL IN QUEUE
      22 18 0086 225      BGEQ 110$                  : BR IF NOT
53    63 D0 0088 226      MOVL (R3),R3                  : FLINK ON TO NEXT ACB.
      F1 11 008B 227      BRB 60$                      :
      008D 228      :
      008D 229      : THE NEW AST IS A NORMAL AST. IT WILL GO AFTER ALL SPECIAL KERNEL ASTS
      008D 230      : AND ASTS WITH LOWER ACCESS MODE.
      008D 231      :
50    OB A5 FC 8F 8B 008D 232 70$: BICB3 #^C<3>,ACB$B_RMOD(R5),R0; GET AST MODE
      53 51 D1 0093 233 80$: CMPL R1,R3                  : CHECK FOR END OF QUEUE
      12 13 0096 234      BEQL 110$                    : INSERT IF AT END
      02 00 ED 0098 235      CMPZV #ACB$V_MODE,#ACB$S_MODE,- :
50    OB A3 236      ACB$B_RMOD(R3),R0                  : COMPARE ACCESS MODES
      05 14 009E 237      BGTR 100$                    : IF GTR AT RIGHT PLACE
53    63 D0 00A0 238 90$: MOVL (R3),R3                  : FLINK ON TO NEXT ACB
      EE 11 00A3 239      BRB 80$                      :
      OB A3 95 00A5 240 100$: TSTB ACB$B_RMOD(R3)        : IS THIS ENTRY A SPECIAL KAST?
      F6 19 00A8 241      BLSS 90$                     : YES, MUST GO AFTER THIS
      00AA 242      :
      00AA 243      : NOW THE CORRECT POSITION HAS BEEN LOCATED. INSERT THE AST CONTROL BLOCK
      00AA 244      : ON THE QUEUE AND COMPUTE THE NEW VALUE FOR ASTLVL BY INTERROGATING THE
      00AA 245      : MODE OF THE AST CONTROL BLOCK AT THE HEAD OF THE QUEUE.
      00AA 246      :
      04 B3 65 0E 00AA 247 110$: INSQUE (R5),@ACB$L_ASTQBL(R3) : INSERT AFTER PREVIOUS
      50 50 D4 00AE 248      CLRL R0                    : ASSUME KERNEL MODE
      51 10 A4 D0 00B0 249      MOVL PCB$L_ASTQFL(R4),R1 : GET HEAD OF AST QUEUE
      OB A1 95 00B4 250      TSTB ACB$B_RMOD(R1)        : IS IT KAST?
      86 19 00B7 251      BLSS 10$                     : BR IF YES TO SET ASTLVL
50    OB A1 FC 8F 8B 00B9 252      BICB3 #^C<3>,ACB$B_RMOD(R1),R0; GET AST MODE FOR HEAD OF QUEUE
      FF7D 31 00BF 253      BRW 10$                     : GO SET ASTLVL
      00C2 254
      00C2 255      .DSABL LSB
      00C2 256      ASSUME ACB$V_MODE EQ 0
      00C2 257      ASSUME ACB$S_MODE EQ 2
      00C2 258      ASSUME ACB$V_KAST EQ 7
      00C2 259
```

```
00C2 261      .SBTTL  MPSSASTSCHEDCHK - CHECK FOR RESCHEDULE AT AST DELIVERY
00C2 262      :++
00C2 263      : FUNCTIONAL DESCRIPTION:
00C2 264      :
00C2 265      : MPSSASTSCHEDCHK is entered to check whether the process that the
00C2 266      : primary was running, should be rescheduled to run on the secondary.
JOC2 267      :
00C2 268      : ENVIRONMENT:
00C2 269      :
00C2 270      : Executed by the primary processor.
00C2 271      : Hooked in at MPH$ASTDELHK and sometimes returns to MPH$ASTDELCONT.
00C2 272      :
00C2 273      :--
00C2 274      :
00C2 275      .ALIGN  LONG
00C4 276 MPSSASTSCHEDCHK::
55 10 B4 0F 00C4 277 REMQUE @PCBSL_ASTQFL(R4),R5 ; Remove head of queue
06 1D 00C8 278 BVS 20$ ; Br if queue empty
00000000'9F 17 00CA 279 10$: JMP @#MPH$ASTDELCONT ; Return to normal code
00 0000'CF 00 E6 00D0 280 20$: BBSSI #LCK$V INTERLOCK,W^MPSS$GL INTERLOCK,30$ ; Flush cache queue
01 0000'CF 03 D1 00D6 281 30$: CMPL W^MPSS$GL_STATE,#MPSS$K_IDLESTATE ; Is secondary idle?
03 12 00DB 282 BNEQ 40$ ; Br on no, dont bother to reschedule
14 03 DA 00DD 283 SOFTINT #3 ; Request a reschedule interrupt
00000000'9F 17 00E0 284 40$: JMP @#MPH$QEMPTYCONT ; Go set null ast level
```



```
00E6 286 .SBTTL MPSSASTNEWLVL - CHECK FOR SETTING NEW ASTLVL
00E6 287 :++
00E6 288 : FUNCTIONAL DESCRIPTION:
00E6 289 :
00E6 290 : MPSSASTNEWLVL is entered to check whether a new AST level should
00E6 291 : be placed in the processor ASTLVL register. If the AST level is kernel,
00E6 292 : it should be left alone. Otherwise, the special indication of an idle
00E6 293 : secondary may be used.
00E6 294 :
00E6 295 : ENVIRONMENT:
00E6 296 :
00E6 297 : Executed by the primary processor.
00E6 298 : Hooked in at MPH$NEWLVL, replaces code from hook to end of routine.
00E6 299 :
00E6 300 :--
00E6 301 :
00E6 302 .ALIGN LONG
00E8 303 MPSSASTNEWLVL::
00E8 304 MTPR R2,#PRS_ASTLVL ; Set ASTLVL register
00EB 305 BEQL 20$ ; If kernel, dont change it
00ED 306 BBSSI #LCK$V_INTERLOCK,W^MPSS$GL_INTERLOCK,10$ ; Flush cache queue
00F3 307 10$: CMPL W^MPSS$GL_STATE,#MPSS$K_IDLESTATE ; Is secondary idle?
00F8 308 BNEQ 20$ ; Br on no, dont change ASTLVL
00FA 309 BITL #^X7FFFFFFF,G^SCH$GL_COMQS ; Is any process other than null COM?
0105 310 BEQL 20$ ; Br on no, nothing else can run
0107 311 PUSHAB PCB$ASTQFL(R4) ; Are there any AST's outstanding
010A 312 CMPL (SP)+,PCB$ASTQFL(R4) ; for this process? If so, don't
010E 313 BNEQ 20$ ; cause a reschedule AST.
0110 314 TSTL PHD$MPINHIBIT(R0) ; Any reason not to run on secondary?
0114 315 BNEQ 20$ ; Br on yes, don't cause reschedule AST
0116 316 MTPR #PSL$C_EXEC,#PRS_ASTLVL ; Set ASTLVL to cause reschedule
0119 317 20$: MOV B R2,PHD$B_ASTLVL(R0) ; Set ASTLVL in PHD
011E 318 ENBINT ; Enable system events again
011E 319 MTPR (SP)+,S^#PRS_IPL ; Return
0121 320 RSB
0122 321 .END
```

13 52 DA 00E8 304
00 0000'CF 00 E6 00EB 305
01 0000'CF D1 00ED 306
00000000'GF 7FFFFFFF 8F D3 00F3 307
10 A4 8E D1 010A 312
0104 C0 D5 0110 314
00CF C0 52 90 0119 317
12 8E DA 011E 318
05 0121 319
0122 320
0122 321

MPAST
Symbol table

- MULTIPROCESSOR AST ROUTINES

B 14

16-SEP-1984 01:59:39 VAX/VMS Macro V04-00
5-SEP-1984 02:06:02 [MP.SRC]MPAST.MAR;1

Page 9
(1)

ACBSB_RMOD	=	0000000B		
ACBSL_ASTQBL	=	00000004		
ACBSL_PID	=	0000000C		
ACBS\$MODE	=	00000002		
ACBSV_KAST	=	00000007		
ACBSV_MODE	=	00000000		
ASTEXIT	=	00000000		
EVT\$ AST	*****		X	02
EXES\$DEANONPAGED	*****		X	02
IPL\$ SYNCH	=	00000008		
LCK\$V INTERLOCK	=	00000000		
MPH\$ASTDELCONT	*****		X	02
MPH\$QEMPTYCONT	*****		X	02
MP\$ASTNEWLVL	000000E8		RG	02
MP\$ASTSCHDCHK	000000C4		RG	02
MP\$SGL_INTERLOCK	*****		X	02
MP\$SGL_STATE	*****		X	02
MP\$INTSCND	*****		X	02
MP\$K_IDLESTATE	=	00000001		
MP\$QAST	00000010		RG	02
PCBSL_ASTQFL	=	00000010		
PCBSL_PHD	=	0000006C		
PCBSL_PID	=	00000060		
PCBSW_STATE	=	0000002C		
PHD\$B_ASTLVL	=	000000CF		
PHD\$M_PINHIBIT	=	00000104		
PR\$ASTLVL	=	00000013		
PR\$IPL	=	00000012		
PR\$SIRR	=	00000014		
PSL\$C_EXEC	=	00000001		
QEXIT	00000066		R	02
QNONEXPR	00000000		R	02
SCH\$C_CUR	=	0000000E		
SCH\$GL_COMQS	*****		X	02
SCH\$GL_CURPCB	*****		X	02
SCH\$GL_PCBVEC	*****		X	02
SCH\$RSE	*****		X	02
SS\$NONEXPR	=	000008E8		
SS\$NORMAL	=	00000001		

! Psect synopsis !

PSECT name	Allocation	PSECT No.	Attributes														
. ABS .	00000000 (0.)	00 (0.)	NOPIC	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE				
\$ABSS	00000000 (0.)	01 (1.)	NOPIC	USR	CON	ABS	LCL	NOSHR	EXE	RD	WRT	NOVEC	BYTE				
\$EXENONPAGED	00000122 (290.)	02 (2.)	NOPIC	USR	CON	REL	LCL	NOSHR	EXE	RD	WRT	NOVEC	LONG				

+-----+
! Performance indicators !
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	47	00:00:00.10	00:00:00.79
Command processing	164	00:00:00.99	00:00:06.15
Pass 1	301	00:00:08.98	00:00:26.48
Symbol table sort	5	00:00:01.42	00:00:02.56
Pass 2	73	00:00:01.74	00:00:05.31
Symbol table output	5	00:00:00.07	00:00:00.21
Psect synopsis output	2	00:00:00.03	00:00:00.09
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	599	00:00:13.33	00:00:41.59

The working set limit was 1500 pages.

47761 bytes (94 pages) of virtual memory were used to buffer the intermediate code.

There were 50 pages of symbol table space allocated to hold 871 non-local and 18 local symbols.

326 source lines were read in Pass 1, producing 14 object records in Pass 2.

23 pages of virtual memory were used to define 22 macros.

+-----+
! Macro library statistics !
+-----+

Macro library name	Macros defined
_\$255\$DUA28:[MP.OBJ]MP.MLB;1	4
_\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	9
_\$255\$DUA28:[SYSLIB]STARLET.MLB;2	7
TOTALS (all libraries)	20

1040 GETS were required to define 20 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:MPAST/OBJ=OBJ\$:MPAST MSRC\$:MPPREFIX/UPDATE=(ENH\$:MPPREFIX)+MSRC\$:MPAST/UPDATE=(ENH\$:MPAST)+EXECML\$/LIB+LIB\$:MP.MLB/LI

0247 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

